

# APPLE II<sup>®</sup> BASIC

---

## QUICK REFERENCE GUIDE

by Gilbert Held



## REFERENCE GUIDE NOTATIONS AND FORMAT CONVENTIONS

A standard scheme for presenting the general format of BASIC language statements is employed in this reference guide. The capitalization, punctuation and other conventions are listed below:

- [ ] Brackets indicate that the enclosed items are optional. Brackets do not appear in the actual statements.
- { } Braces indicate that a choice of one of the enclosed items is to be made. Braces do not appear in the actual statements.
- ... Ellipses indicate that the preceding item may be repeated. Ellipses do not appear in the actual statements.

**Italics** Italics indicate generic terms. The programmer must supply the

actual value or wording required. See Generic Terms and Abbreviations.

**Line number** A line number is implied for all BASIC language statements.

**Punctuation** All punctuation characters, including commas, semicolons, colons, quotation marks and parentheses, must appear as indicated.

**UPPER CASE** Upper case letters and words must appear exactly as indicated.

## APPLE BASIC VERSIONS

**Applesoft**—Comprehensive BASIC which permits numeric values with fractions. Line numbers must be between 0 and 63999.

**Integer BASIC**—A version of BASIC which limits calculated values to a range of 32767 to -32767, and does not permit decimal frac-

tions. Line numbers must be between 0 and 32767.

### VIDEO PROMPT CHARACTERS

Character	Language
*	Monitor
>	Integer BASIC
]	Applesoft

## GENERIC TERMS AND ABBREVIATIONS

**Arg**—Argument.

**Bn**—A byte number.

**col**—Low resolution graphics column number.  $0 \leq \text{numeric expression} \leq 39$ .

**colh**—High resolution graphics column number.  $0 \leq \text{numeric expression} \leq 279$ .

**const**—Any numeric or string constant.

**Dn**—A disk drive number. Must be specified as D0 or D1.

**expr**—Any valid Apple II expression.

**expr\$**—Any valid Apple II string constant, variable or expression.

**exprnm**—Any numeric constant, variable or expression.

**filename**—A disk file name.

**format**—Format of BASIC language statement common to both Applesoft and Integer BASIC.

**format<sub>AS</sub>**—Applesoft BASIC format of BASIC language statement.

**format<sub>IB</sub>**—Integer BASIC format of BASIC language statement.

**length**—Integer ranging from 0 through 32767 decimal.

**line**—A BASIC program line number.

**line<sub>i</sub>**—The *i*th line number in a BASIC program.

**Ln**—Length in bytes. Integer ranging in value from 0 to 32767 decimal.

**memadr**—The memory address referenced by a numeric expression, variable or constant.  $-65535_{10} \leq \text{range} \leq 65535_{10}$ .

**memloc**—the memory location specified by an integer constant.  $0_{10} \leq \text{range} \leq 65535_{10}$ .

**msg**—The message denoted by quotes which forms a string.

**nvar**—Numeric variable.

**row**—Low resolution graphics row number.  $0 \leq \text{numeric expression} \leq 47$ .

**rowh**—High resolution graphics row number.  $0 \leq \text{numeric expression} \leq 191$ .

**Rn**—Specified field of a file.

**slot**—predefined areas into which circuit boards plug.

**Sn**—Slot number for input or output.  $0 \leq n \leq 7$ .

**sub**—Subscript.

**var**—Numeric or string variable in Integer BASIC. Numeric, integer or string variable in Applesoft.

**varnm**—A numeric variable name.

**var(sub)**—Any subscripted numeric variable in Integer BASIC. Any subscripted integer, numeric or string variable in Applesoft.

**Vn**—A disk volume number.  $0 \leq n \leq 255$ .

## VARIABLE NAMING CONVENTIONS

### Applesoft format F{SL} where:

**F** First character must be a letter.

**S** Second character is optional and may be a letter or digit.

**L** Last character denotes variable type: \$ for string variable, % for integer variable, otherwise real variable.

Name length must be 1, 2 or 3 characters.

### Integer BASIC format F{SL} where:

**F** First character must be a letter.

**S** Second character to last character may be letter(s) or digit(s).

**L** Last character must be a \$ for string variable.

Name length may be 1 to 100 characters.

## ARITHMETIC OPERATORS

OPERATION	INTEGER BASIC	APPLESOFT
ARITHMETIC		
Exponentiation	^	^
Unary Minus	-	-
Multiplication	*	*
Division	/	/
Division Remainder	MOD	Not Available
Addition	+	+
Subtraction	-	-
RELATIONAL		
Equal	=	=
Not Equal	#	<> or ><
Less than	<	<
Greater than	>	>
Less than or equal	<=	<= or =<
Greater than or equal	>=	>= or =>
BOOLEAN		
Logical complement	NOT	NOT
Logical AND	AND	AND
Logical OR	OR	OR

## SYSTEM COMMANDS

**AUTO**—Integer BASIC command that sets the automatic line numbering mode.

format<sub>IB</sub>: AUTO line [, increment ]

**CLR**—Integer BASIC command that assigns zero values to all variables and array elements.

format<sub>IB</sub>: CLR

**CON**—Integer BASIC command that causes a program to resume execution at the next instruction following a halt.

format<sub>IB</sub>: CON

**CONT**—Applesoft command that causes a program to resume execution at the next instruction following a halt.

format<sub>AS</sub>: CONT

**DEL**—Eliminates the specified

program lines.

format: DEL line<sub>1</sub>, line<sub>2</sub>

**FP**—Puts the Apple II into Applesoft.

format: FP[,Dn] [,Sn] [,Vn]

**INT**—Puts the Apple II into Integer BASIC.

format: INT

**LIST**—Displays all or a specified portion of the program currently in memory.

format: LIST line<sub>1</sub> [, line<sub>2</sub>]

format<sub>AS</sub>: LIST { line<sub>1</sub> [{ ; }]  
[line<sub>1</sub>]{ ; } line<sub>2</sub> }

**MAN**—Ends the automatic line numbering mode in Integer BASIC.

format<sub>IB</sub>: MAN

**MON**—Permits the display of disk commands and/or data traffic on the video monitor.



format: MON [C] [,I] [,O]

where: C is specified to display disk commands

I is specified to display data flow from disk to Apple II

O is specified to display data flow from Apple II to the disk

**NEW**—Deletes the current program and all variables from memory in the Apple II.

format: NEW

**NOMON**—Ends the video display of

disk commands and/or data flow initiated by MON command.

format: NOMON

**RUN**—Executes the current program in memory. If a line number is specified, execution commences at that number, otherwise execution begins at the lowest numbered line in the program.

format: RUN [*line*]

**SAVE**—Saves the program currently in memory on cassette or on disk. See Input/Output control for formats.

## BASIC LANGUAGE STATEMENTS

### BRANCHING

**CALL**—Results in a branch to a machine language subroutine at a specified memory location.

format: CALL *memadr*

**GOSUB**—Results in a branch to the indicated *line* number. A return statement causes a branch back to the instruction following the GOSUB.

format: GOSUB *line*

**GOTO**—Causes an unconditional branch to the indicated *line* number or the numeric expression in Integer BASIC.

format: GOTO *line*

format: <sub>IB</sub>: GOTO *exprnm*

**IF-THEN**—Causes a branch or the execution of a statement to occur if the indicated expression is true.

format: IF *expr* THEN *statement*  
[:*statement* ...]

$$\text{IF } \textit{expr} \left\{ \begin{array}{l} \text{THEN} \\ \text{GOTO} \\ \text{THEN GOTO} \end{array} \right\} \textit{line}$$

format: <sub>IB</sub>: IF *expr* THEN *statement*  
IF *expr* THEN [GOTO] *line*

**ON-GOSUB**—Causes a conditional Applesoft subroutine call. Branch depends upon the computed or current value of the expression.

format: <sub>AS</sub>: ON *exprnm* GOSUB  
*line* [, *line* ...]

**ON-GOTO**—Causes a conditional Applesoft branch based upon the current or computed value of the expression.

format: <sub>AS</sub>: ON *exprnm* GOTO *line*  
[, *line* ...]

**POP**—Causes the most recently executed GOSUB to be changed to a GOTO by removing the return location for the last GOSUB statement executed.

format: POP

**RETURN**—Results in a program branch to the statement immediately following the most recently executed GOSUB statement.

format: RETURN

### ERROR DETERMINATION AND CONTROL

**DSP**—Integer BASIC statement that displays the changing values of a predefined variable during program execution.

format: <sub>IB</sub>: DSP *var*

**NO DSP**—Integer BASIC statement that cancels the display mode for a predefined variable.

format: <sub>IB</sub>: NO DSP *var*

**NO TRACE**—Turns off the program execution TRACE initiated by a TRACE statement.

format: NO TRACE

**ONERR GOTO**—Applesoft statement that causes a branch to the *line* number indicated if an error occurs in the program.

format: <sub>AS</sub>: ONERR GOTO *line*

**RESUME**—Applesoft statement that causes the program to resume at the beginning of an instruction in which an error previously occurred.

format: <sub>AS</sub>: RESUME

**TRACE**—Displays the line number of each statement executed.

format: TRACE

### MEMORY REFERENCE

**CLEAR**—Applesoft statement that initializes all numeric variables and array elements to zero and

assigns a null value to all strings.

format: <sub>AS</sub>: CLEAR

**HIMEM**—Establishes the highest location in RAM memory available for use by the BASIC program.  
format: HIMEM: *exprnm*

**LOMEM**—Establishes the lowest location in RAM memory available for use by the BASIC program.  
format: LOMEM: *exprnm*

### PROCESSING STATEMENTS

**DATA**—Applesoft statement that creates a list of values to be assigned to variables through the use of a READ statement.  
format: DATA *const* [, *const* ...]

**DEF FN**—Applesoft statement that permits special purpose functions to be defined.  
format: DEF FN*var* (*arg*) = *exprnm*

**DIM**—Reserves space in main memory for an array or string.  
format<sub>AS</sub>: DIM *var*(*sub*[*sub* ...])  
[, *var*(*sub*[*sub* ...]) ...]  
format<sub>IB</sub>: DIM *var*(*sub*) [, *var*(*sub*) ...]

**END**—Causes the program to terminate.  
format: END

**FOR**—Initiates a loop that repeats execution of all instructions bounded by the NEXT statement until the automatically incremented variable attains the value *exprnm*<sub>2</sub>.  
format: FOR *varnm* = *exprnm*<sub>1</sub>,  
TO *exprnm*<sub>2</sub> [STEP *exprnm*<sub>3</sub>]

**LET**—Assigns a value to the specified variable.  
format: [LET] *var* = *expr*

**NEXT**—Terminates the loop initiated by a FOR statement  
format: NEXT *varnm* [, *varnm* ...]

**READ**—Applesoft statement that assigns values from DATA statements to variables.  
format<sub>AS</sub>: READ *var* [, *var* ...]

**REM**—Nonexecutable statement that permits remarks to be placed in the program.  
format: REM *remark*

**RESTORE**—Applesoft statement that resets the DATA list pointer to the beginning of the list.  
format<sub>AS</sub>: RESTORE

**STOP**—Applesoft statement that causes the program to halt execution.  
format<sub>AS</sub>: STOP

**WAIT**—Applesoft statement that halts a program until a specified memory location attains a defined value.  
format<sub>AS</sub>: WAIT *memadr*, *exprnm*,  
[, *exprnm*]

## INPUT/OUTPUT CONTROL

### GENERAL

**GET**—Applesoft statement that receives one character from the keyboard without displaying the character.  
format<sub>AS</sub>: GET *var*

**IN#**—Selects the peripheral *slot* from which subsequent input will occur.  
format: IN# *slot*

**INPUT**—Optionally displays a *prompt message* and then accepts input data, assigning values to the variables listed.  
format: INPUT ["*prompt message*";] *var* [, *var* ...]

**PDL**—Function that returns the current value of the game

control device.  
format: PDL (*exprnm*)

**PR#**—Selects the peripheral *slot* which will receive subsequent output.  
format: PR# *slot*

**PRINT**—Outputs values to the display or to other designated devices.  
format: [*const*, *expr*, *expr*\$,  
*exprnm*, *varnm*, *var*(*sub*)]

**TAB**—Integer BASIC statement that positions the cursor to the specified column at the current line.  
format<sub>IB</sub>: TAB *col*

### CASSETTE CONTROL

**LOAD**—Loads the program on cassette into memory.  
format: LOAD

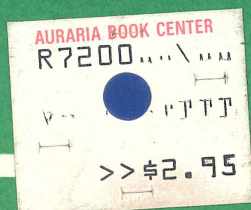
**RECALL**—Applesoft statement that retrieves a numeric array from cassette storage.  
format<sub>AS</sub>: RECALL *varnm*

**SAVE**—Saves the program currently in memory on cassette storage.  
format: SAVE

**SHLOAD**—Applesoft statement that loads a high-resolution graphics shape table from cassette storage.  
format<sub>AS</sub>: SHLOAD

**STORE**—Applesoft statement that saves a specified array on cassette storage.  
format<sub>AS</sub>: STORE *varnm*





## DISK CONTROL

**APPEND**—Opens a disk file, positioning the file pointer at the end of the file.

\*format: APPEND *filename* [, *Dn*] [, *Sn*] [, *Vn*]

**BLOAD**—Loads a binary disk file starting at the memory location specified.

\*format: BLOAD *filename* [, *Amemloc*] [, *Dn*] [, *Sn*] [, *Vn*]

**BRUN**—Retrieves, loads and executes a machine language (binary) program.

\*format: BRUN *filename* [, *Amemloc*] [, *Dn*] [, *Sn*] [, *Vn*]

**BSAVE**—Saves a *length* of Apple II's memory in binary on designated disk file.

\*format: BSAVE *filename*, *Amemloc*, *L length* [, *Dn*] [, *Sn*] [, *Vn*]

**CATALOG**—Provides a list of all files on the specified disk.

\*format: CATALOG [, *Dn*] [, *Sn*]

**CHAIN**—Integer BASIC statement that loads and executes a program from disk.

\*format: CHAIN *filename* [, *Dn*] [, *Sn*] [, *Vn*]

**CLOSE**—Closes all open disk files or a specified open disk file.

\*format: CLOSE [*filename*]

**DELETE**—Erases designated file from the disk.

\*format: DELETE *filename* [, *Dn*] [, *Sn*] [, *Vn*]

**EXEC**—Permits designated text file to control Apple II.

\*format: EXEC *filename* [, *Rn*] [, *Dn*] [, *Sn*] [, *Vn*]

**FP**—Retrieves Applesoft from disk.

format: FP [, *Dn*] [, *Sn*] [, *Vn*]

**INIT**—Initializes disk by saving program in memory on disk under designated *filename*.

\*format: INIT *filename* [, *Dn*] [, *Sn*] [, *Vn*]

**LOAD**—Loads program with designated *filename* from disk.

\*format: LOAD *filename* [, *Dn*] [, *Sn*] [, *Vn*]

**LOCK**—Protects a disk file from deletion until it is unlocked.

\*format: LOCK *filename* [, *Dn*] [, *Sn*] [, *Vn*]

**MAXFILES**—Permits up to 16 disk files to be open.

\*format: MAXFILES *limit*

**OPEN**—Prepares a disk file for access.

\*format: OPEN *filename* [, *Dn*] [, *Sn*] [, *Vn*]

**POSITION**—Positions the disk file pointer *Rn* fields in front of its current position.

\*format: POSITION *filename* [, *Rn*]

**PR#**—Selects designated peripheral slot to receive output.

\*format: PR# *slot*

**READ**—Designates a disk file from which subsequent INPUT and GET commands receive data.

\*format: READ *filename* [, *Rn*] [, *Bn*]

**RENAME**—Changes the name of a disk file without changing its contents.

\*format: RENAME *filename*<sub>1</sub>, *filename*<sub>2</sub> [, *Dn*] [, *Sn*] [, *Vn*]

**RUN**—Loads and executes a program residing on disk storage.

\*format: RUN *filename* [, *Dn*] [, *Sn*] [, *Vn*]

**SAVE**—Permanently stores program on disk with designated *filename*.

\*format: SAVE *filename* [, *Dn*] [, *Sn*] [, *Vn*]

**UNLOCK**—Permits file to be changed or deleted by removing its locked status.

\*format: UNLOCK *filename* [, *Dn*] [, *Sn*] [, *Vn*]

**VERIFY**—Checks to insure file saved correctly on disk.

\*format: VERIFY *filename* [, *Dn*] [, *Sn*] [, *Vn*]

**WRITE**—Designates the disk file to which subsequent PRINT statements output data.

\*format: WRITE *filename* [, *Rn*] [, *Bn*]

\*Indicates Disk Operating System (DOS) commands which require a PRINT statement containing a CTRL-D character in the programmed mode operation.

## VIDEO DISPLAY COLOR CONTROLS

## COMPLEMENT COLORS

COLOR	XDRAW COLOR	COLOR	XDRAW COLOR
Black	White	Orange	Blue
White	Black	Green	Violet
Violet	Green	Blue	Orange

### LOW RESOLUTION GRAPHICS COLORS

COLOR	NUMBER	COLOR	NUMBER
Black	0	Brown	8
Magenta	1	Orange	9
Dark Blue	2	Gray #2	10
Purple	3	Pink	11
Dark Green	4	Light Green	12
Gray #1	5	Yellow	13
Medium Blue	6	Aqua	14
Light Blue	7	White	15

### HIGH RESOLUTION GRAPHICS COLORS

COLOR	NUMBER
Black	0
Green	1
Violet	2
White	3
Black	4
Orange	5
Blue	6
White	7

## VIDEO AND GRAPHICS CONTROL

**COLOR**—Sets the low-resolution graphics color.

format: COLOR = *exprnm*

**DRAW**—Applesoft statement that draws a high-resolution graphics shape on the video display.

format<sub>AS</sub>: DRAW *exprnm* [AT *colh, rowh*]

**FLASH**—Applesoft statement that flashes the video display output.

format<sub>AS</sub>: FLASH

**GR**—Puts display into low-resolution graphics mode of 40 col × 40 rows.

format: GR

**HCOLOR**—Applesoft statement that sets the color for plotting in the high-resolution graphics mode.

format<sub>AS</sub>: HCOLOR = *exprnm*

**HGR**—Applesoft statement that puts the video display into the high-resolution graphics mode of 280 columns × 160 rows, with a four-line text window.

format<sub>AS</sub>: HGR

**HGR2**—Sets the display to a full-screen, high-resolution graphics mode of 280 columns × 192 rows.

format: HGR2

**HLIN**—Draws a horizontal line on the display from *col<sub>1</sub>* to *col<sub>2</sub>* at the specified *row* in low-resolution graphics.

format: HLIN *col<sub>1</sub>, col<sub>2</sub>* AT *row*

**HOME**—Applesoft statement that clears the display and positions the cursor at row 1, column 1.

format<sub>AS</sub>: HOME

**HPLOT**—Applesoft statement that places a dot or draws a line on the display in the high-resolution graphics mode.

formats<sub>AS</sub>:

HPLOT *colh, row*

HPLOT TO *colh, rowh*

HPLOT *colh<sub>1</sub>, rowh<sub>1</sub>* TO *colh<sub>2</sub>, rowh<sub>2</sub>*  
[TO *colh<sub>3</sub>, rowh<sub>3</sub>* . . .]

**HTAB**—Applesoft statement that moves the cursor to the specified column.

format<sub>AS</sub>: HTAB *col*

**INVERSE**—Applesoft statement that puts display output into the inverse video mode.

format<sub>AS</sub>: INVERSE

**NORMAL**—Applesoft statement that turns off the FLASH and INVERSE video modes.

format<sub>AS</sub>: NORMAL

**PLOT**—Displays a point on the low-resolution graphics display or a colored character in the text window.

format: PLOT *col, row*

**POS**—Function that returns the column position of the cursor.

format: POS (*exprnm*)

**ROT**—Applesoft statement that rotates high-resolution shapes drawn by DRAW or XDRAW.

format<sub>AS</sub>: ROT = *exprnm*

**SCALE**—Applesoft statement that sets the size of high-resolution shapes drawn by DRAW or XDRAW.

format<sub>AS</sub>: SCALE = *exprnm*

**SCRN**—Function that returns the color code of the low-resolution graphics point at the specified coordinates.

format: SCRN (*col, row*)

**SPC**—Function that moves the cursor a specified number of positions to the right.

format: SPC (*exprnm*)

**SPEED**—Applesoft statement that changes the rate at which characters are output to the display.

format: SPEED *exprnm*



**TEXT**—Returns display to the full-screen text mode from any graphics mode.

format: TEXT

**VLIN**—Draws a vertical line from  $row_1$  to  $row_2$  in column  $col$  on the display in the low-resolution graphics mode.

format: VLIN  $row_1$ ,  $row_2$  AT  $col$

**VTAB**—Positions the cursor to the line specified by  $row$  in the current display column.

format: VLIN  $row$

**XDRAW**—Applesoft statement that draws shape  $exprnm$  at the designated location.

format<sub>AS</sub>: XDRAW  $exprnm$  [AT  $colh$ ,  $rowh$ ]

## BASIC FUNCTIONS

### FUNCTION FORMAT AND DESCRIPTION

**ABS ( $exprnm$ )**—Returns the absolute value of a number.

**ASC ( $expr$$ )**—Returns the ASCII code number for the specified character.

**ATN ( $exprnm$ )**—Returns the arctangent of an angle of  $exprnm$  radians.

**CHR\$ ( $exprnm$ )**—Returns the character (string value) of the specified ASCII code.

**COS ( $exprnm$ )**—Returns the cosine of an angle of  $exprnm$  radians.

**EXP ( $exprnm$ )**—Returns the base of the natural logarithm (e) raised to the specified power.

**FN  $varnm$  ( $exprnm$ )**—Calls the previously defined function  $varnm$  with the value  $exprnm$  assigned.

**FRE ( $exprnm$ )**—Returns the number of bytes of memory available for an Applesoft program.

**INT ( $exprnm$ )**—Returns the integer portion of a number.

**LEFT\$ ( $expr$,  $exprnm$ )$** —Returns the leftmost  $exprnm$  characters of the string  $expr$$ .

**LEN ( $expr$$ )**—Returns the length of the specified string.

**LOG ( $exprnm$ )**—Returns the natural logarithm of the specified number.

**MID\$ ( $expr$,  $exprnm_1$ ,  $exprnm_2$ )$** —Returns  $exprnm_2$  characters from  $expr$$ , commencing with character  $exprnm_1$ .

**PDL ( $exprnm$ )**—Returns the current value of the specified game control paddle.

**PEEK ( $memadr$ )**—Returns the decimal value of a specified memory location.

**POS ( $exprnm$ )**—Returns the column position of the cursor.

**RIGHT\$ ( $expr$,  $exprnm$ )$** —Returns the rightmost  $exprnm$  characters of  $expr$$ .

**RND ( $exprnm$ )**—Returns a random number.

**SCRN ( $col$ ,  $row$ )**—Returns the color code of the specified coordinates when in low-resolution graphics mode.

**SGN ( $exprnm$ )**—Returns +1 if  $exprnm$  is positive, -1 if negative, and 0 if its value is zero.

**SIN ( $exprnm$ )**—Returns the sine of an angle of  $exprnm$  radians.

**SPC ( $exprnm$ )**—Moves the cursor  $exprnm$  positions to the right.

**SQR ( $exprnm$ )**—Returns the square root.

**STR\$ ( $exprnm$ )**—Converts a numeric value to a string.

**TAB ( $exprnm$ )**—Used with the PRINT statement to move the cursor to the specified column position.

**TAN ( $exprnm$ )**—Returns the tangent of an angle of  $exprnm$  radians.

**USR ( $exprnm$ )**—Causes a branch to a machine language subroutine indirectly through memory locations 10 through 12, specified by  $exprnm$ .

**VAL ( $exprnm$ )**—Returns the numeric value of a string.

## CONTROL CHARACTER UTILIZATION

**CTRL-B**—Places the Apple into the version of BASIC in Read Only Memory. For the Apple II Plus this places it into Applesoft BASIC.

**CTRL-C**—Halts a listing.

**CTRL-D**—Used in conjunction with appropriate PRINT statements to direct output to, or receive input from, a peripheral device.

**CTRL-G**—Rings the Apple II bell.

**CTRL-H**—Backspace.

**CTRL-J**—Generates a linefeed.

**CTRL-K**—Boots DOS from the

monitor after the appropriate slot number is typed. In Monitor Mode, substitutes the device in a given backplane slot for the Apple keyboard for data input.

**CTRL-M**—Generates a carriage return.

**CTRL-P**—When followed by appropriate slot number, boots DOS from the Monitor and Integer BASIC. In Monitor Mode, diverts all output for the screen to an Apple intelligent interface.

**CTRL-S**—Temporarily suspends the listing of a program until any other key is pressed.